Imperial College London

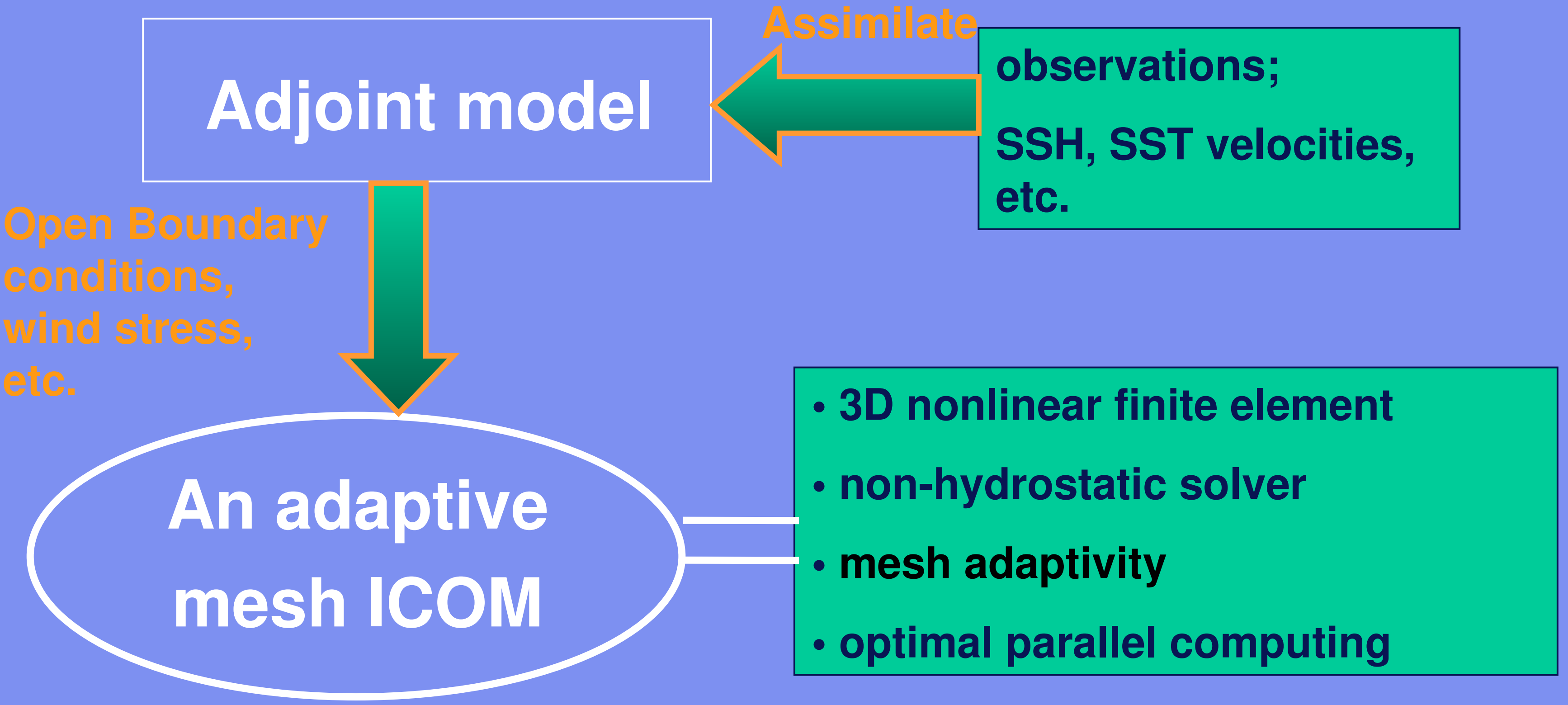
Implementation and application of an adaptive mesh adjoint model with ICOM

Outline

- Objective
- Features of the forward and adjoint models
- Implementation of the adjoint model in ICOM
- Applications
- Discussion and future work

OBJECTIVE

To develop an adjoint model to assimilate observations into a 3D unstructured prognostic model with free surface



Forward model

$$\nabla \cdot \mathbf{u} = 0,$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + f\mathbf{k} \times \mathbf{u} = -\nabla p - \rho g \mathbf{k} + \nabla \cdot \tau$$

$$\frac{\partial \zeta}{\partial t} + \mathbf{u} \cdot \left(\frac{\partial \zeta}{\partial x}, \frac{\partial \zeta}{\partial y}, -1 \right) = 0$$

Euler-Lagrange method

Functional

$$\Im(\mathbf{u},\zeta,\mathbf{m}) = \int_t \int_\Omega \frac{1}{2}\Bigg[\sum_{d_\mathbf{u}=1}^{\mathcal{D}_\mathbf{u}} (\mathbf{u}-\mathbf{u}_{o,d_\mathbf{u}})^T \mathcal{W}_{\mathbf{u},d_\mathbf{u}} (\mathbf{u}-\mathbf{u}_{o,d_\mathbf{u}})\,\delta(\Omega-\Omega_{d_\mathbf{u}}) + \sum_{d_\zeta=1}^{\mathcal{D}_\zeta} \mathcal{W}_{\zeta,d_\zeta} (\zeta-\zeta_{o,d_\zeta})^2\,\delta(\Omega-\Omega_{d_\zeta})\Bigg]\,d\Omega\,dt,$$

Lagrangian Functional

$$\begin{aligned} L(\phi,\phi^*,\mathbf{m}) &= \Im(\mathbf{u},\zeta,\mathbf{m}) + \int_t\int_\Omega (E_1\cdot\mathbf{u}^* + E_2\,p^* + E_3\,\zeta^*\,\delta(z-\zeta))\,d\Omega\,dt \\ &= \Im(\mathbf{u},\zeta,\mathbf{m}) + \\ &\quad \int_t\int_\Omega (\nabla\cdot\mathbf{u})\,p^*\,d\Omega\,dt + \\ &\quad \int_t\int_\Omega \Bigg[\frac{\partial\mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} + f\mathbf{k}\times\mathbf{u} + g\Delta\rho + \\ &\qquad \rho g\left(\frac{\partial\zeta}{\partial x},\frac{\partial\zeta}{\partial y},0\right)^T + \nabla p - \nabla\cdot\tau\Bigg]\cdot\mathbf{u}^*\,d\Omega\,dt + \\ &\quad \int_t\int_\Omega \left[\frac{\partial\zeta}{\partial t} + \mathbf{u}\cdot\left(\frac{\partial\zeta}{\partial x},\frac{\partial\zeta}{\partial y},-1\right)\right]\zeta^*\,\delta(z-\zeta)\,d\Omega\,dt, \end{aligned}$$

Tangent Linear Model

$$\nabla \cdot \bar{\mathbf{u}} = 0,$$

$$\frac{\partial \bar{\mathbf{u}}}{\partial t} + \mathbf{u} \cdot \nabla \bar{\mathbf{u}} + f\mathbf{k} \times \bar{\mathbf{u}} + \bar{\mathbf{u}} \cdot \nabla \mathbf{u} + \rho g \left(\frac{\partial \bar{\zeta}}{\partial x}, \frac{\partial \bar{\zeta}}{\partial y}, 0 \right)^T + \nabla \bar{p} - \nabla \cdot \bar{\tau} = 0,$$

$$\frac{\partial \bar{\zeta}}{\partial t} + \mathbf{u} \cdot \left(\frac{\partial \bar{\zeta}}{\partial x}, \frac{\partial \bar{\zeta}}{\partial y}, 0 \right) + \bar{\mathbf{u}} \cdot \left(\frac{\partial \zeta}{\partial x} \frac{\partial \zeta}{\partial y}, -1 \right) = 0,$$

Features of the forward and adjoint models

- Different options for non-linear discretisation and adaptive meshes for the forward and adjoint models;
- Dynamically adapt the mesh to optimise the accuracy of the inversion problem and forward solution;
- Incorporation of changing computational domain (free surface) into the 3-D adjoint model and sensitivity analysis;
- Inclusion of penalty terms to remove ill-posedness of the inversion problems and regularise control variables spatially and temporarily;
- Potential to accelerate the inversion with a hierarchy of increasingly fine mesh inversions.

Definition of the inverse problem

• **Define a cost function which is the misfit between numerical solution and observations**

$$= (m) \; \frac{1}{2} \int_t \int_\Omega [f(\mathbf{x},\mathbf{x}_o,t) \quad g(\mathbf{x},\mathbf{x}_o,t)] d\Omega \; dt$$

Where, $f(\mathbf{x},\mathbf{x}_o,t)$ **and** $g(\mathbf{x},\mathbf{x}_o,t)$

are the numerical solution and observation at the detector $\mathbf{x}_{o;}$

m **is the control to be optimised, such as, initial and boundary conditions, wind stresses, bottom drag coefficients, permeability, saturations etc.**

Optimization

Minimize the cost function and find an optimal control *m*

Nonlinear conjugate gradient method

$$\mathbf{m}_{i+1} = \mathbf{m}_i + \alpha \mathbf{d}_i ; \mathbf{d}_{i+1} = -\mathbf{g}_i + \beta_i \mathbf{d}_i ; \beta_i = \frac{\mathbf{g}_i^T (\mathbf{g}_{i+1} - \mathbf{g}_i)}{\mathbf{g}_i^T \mathbf{g}_i}$$

Here, an adjoint method is used to calculate the gradient of the cost function (nonlincg.F90).

Adjoint model

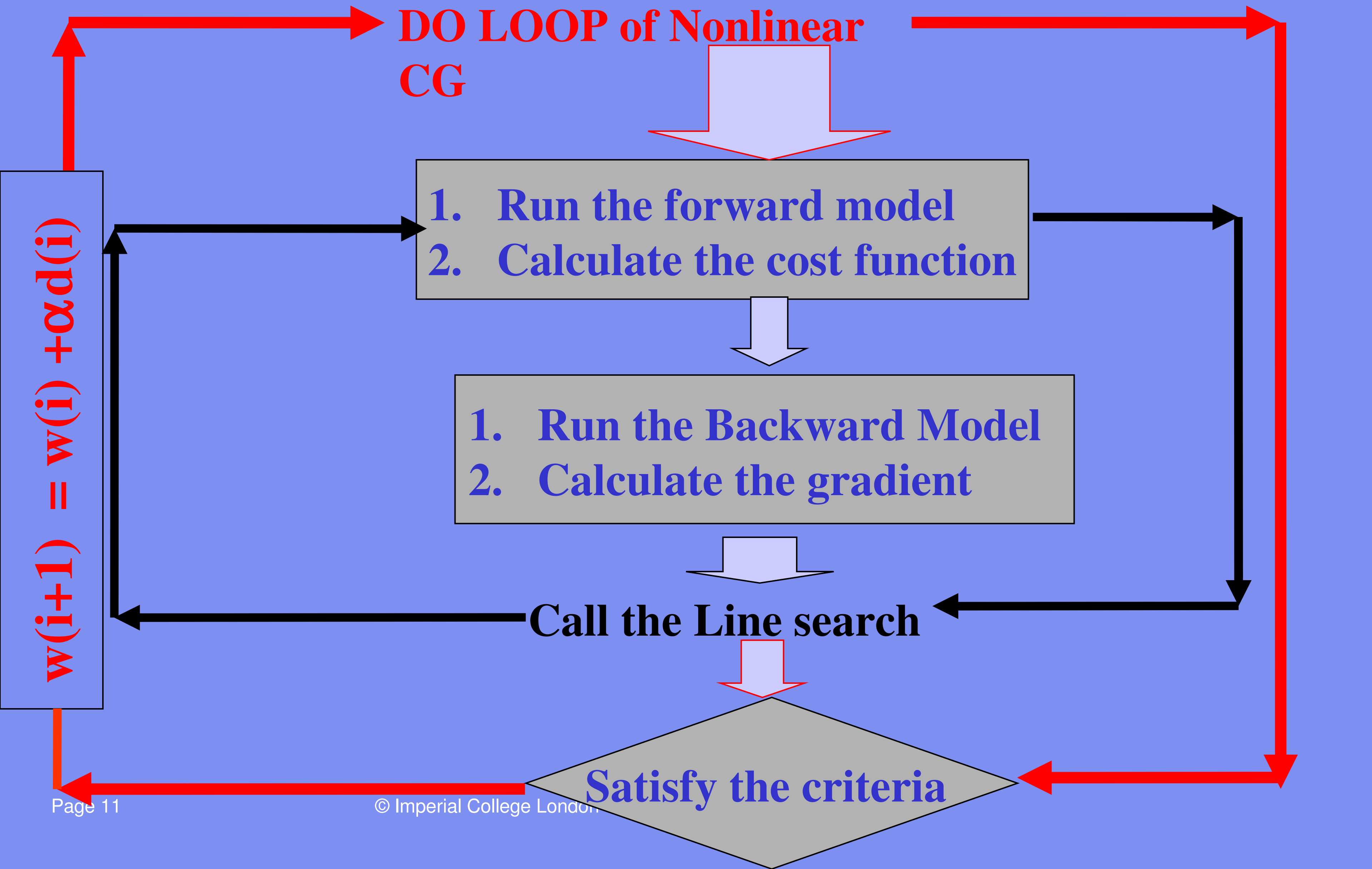
$$-\nabla \cdot \bar{\mathbf{u}}^* = 0,$$

$$-\frac{\partial \mathbf{u}^*}{\partial t} - \mathbf{u} \cdot \nabla \mathbf{u}^* - f\mathbf{k} \times \mathbf{u}^* - \nabla p^* - \nabla \mathbf{u}^{*T} \cdot \mathbf{u}$$

$$-\zeta \cdot \left(\frac{\partial \zeta^*}{\partial x}, \frac{\partial \zeta^*}{\partial y}, 0 \right)^T - (0, 0, \zeta^*)^T - (\nabla \cdot \tau)^* = (S_u, S_v, S_w)^T,$$

$$-\frac{\partial \zeta^*}{\partial t} - u \cdot \left(\frac{\partial \zeta^*}{\partial x}, \frac{\partial \zeta^*}{\partial y}, 0 \right) + \rho g \left(\frac{\partial u^*}{\partial x} + \frac{\partial v^*}{\partial y} \right) + E_{fs} + E\tau^* = S_\zeta.$$

Implementation of the adjoint model in ICOM

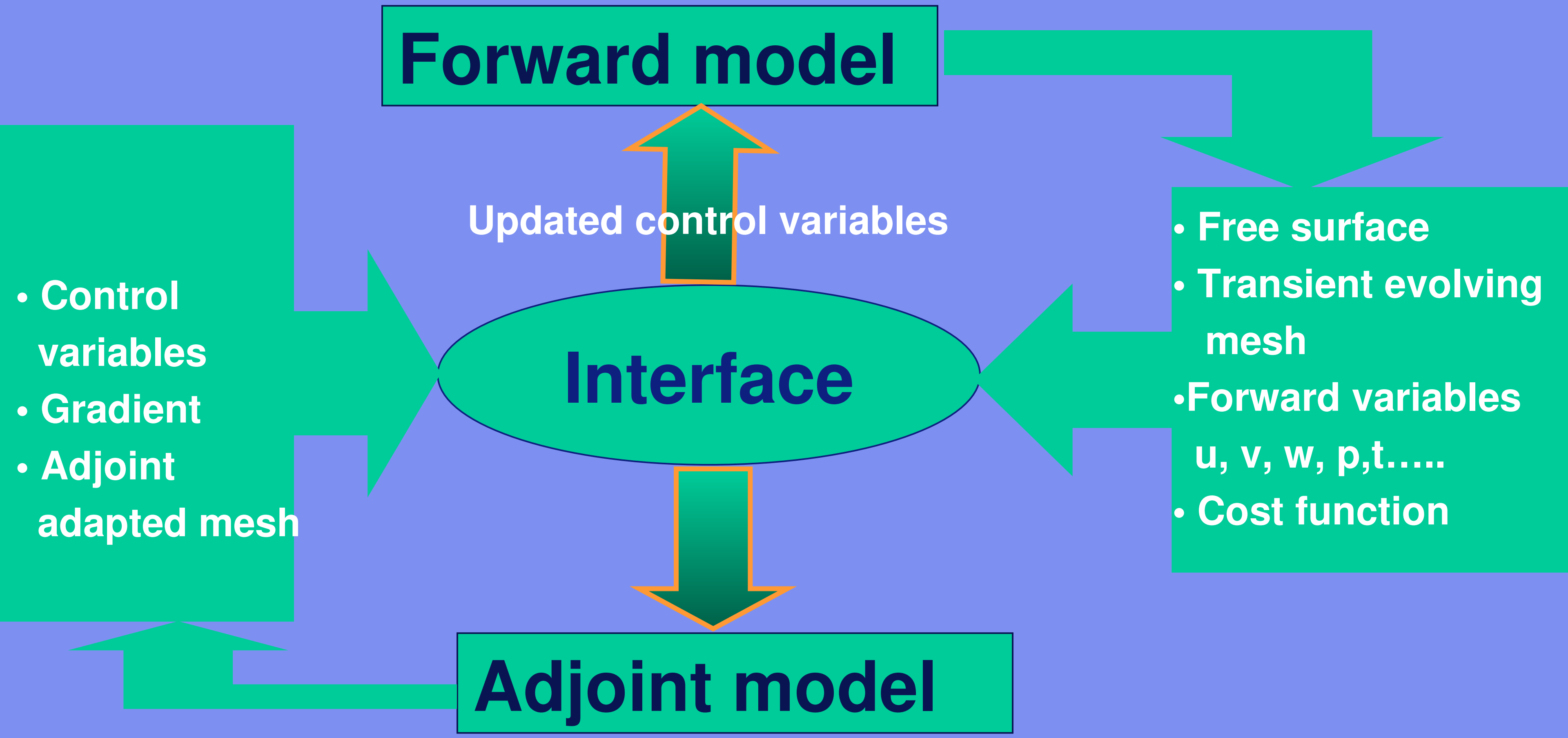


\# → Which comes from Fortran pre-processor --- C pre-processor

→ The compiler can use it to get a different objective file (Makefile & autoconfig)

```
# ifdef FORWARD
    print *, 'Going forward'
    Subroutine Forwardstuff ( )
# ifdef ADJOINT
    Subroutine Adjointstuff ( )
    print *, 'Going backward'
# endif
    ……………
    Common codes for all cases
   ……………
End Program
```

Inversion with a dynamically-adapting mesh



Implementation of the adjoint model in ICOM
---- run the forward model

Example.gem

- **Output at each time level---forward_interface.F90**

  (1) the forward solution;

  (2) the forward meshes;

  (3) the free surface.

- **Calculate the objective function---function.F90;**

Read the observational data: ---- exasol.F90

Implementation of the adjoint model in ICOM

--- Run the adjoint model

Example_ADJ.gem

- **The same computer domain for the forward and adjoint models—interpolate the free surface from the forward mesh onto the adjoint mesh;**
- **Interpolate the forward solutions from the forward mesh onto the adjoint mesh;**
- **Calculate the source terms – sourceFS.F90;**
- **Calculate the previous and current gradients;**

Adjoint_interface.F90,
Readdata_nonl_gradient.F90

Data files for running the adjoint model

Input data (input.dat):

2. Forward and adjoint file names;
3. Observational file name;
4. The numbers of time levels and detectors

Observational data (AdjSourceData.dat):

2. Positions
3. Time levels
4. Observational data at each time levels

Gaussian spreading data (GaussianSpread.dat):

2. Number of the detectors where the observational data is available
3. Width of the Gaussian function

Test cases: Inversion of free surface height along the open boundary for 2D/3D tidal flow

The cost function (when considering assimilation of the sea level)

$$= (\zeta, \zeta_b) \quad \frac{1}{2} \int_t \int_\Omega \sum_{k=1}^{Nos} (\zeta - \zeta_o)^T W_{o,k} (\zeta - \zeta_o) d\Omega \, dt + \frac{1}{2} \lambda \int_t \int_\Omega \zeta_b^T \zeta_b \, d(\Omega) dt$$

The observations are obtained using an identical twin experiment

The water depth: $H_0 = 65m$

The exact inlet tidal height: $\eta_{exact} = 1.0 \sin(t/T); T = 12 \quad 3600s$

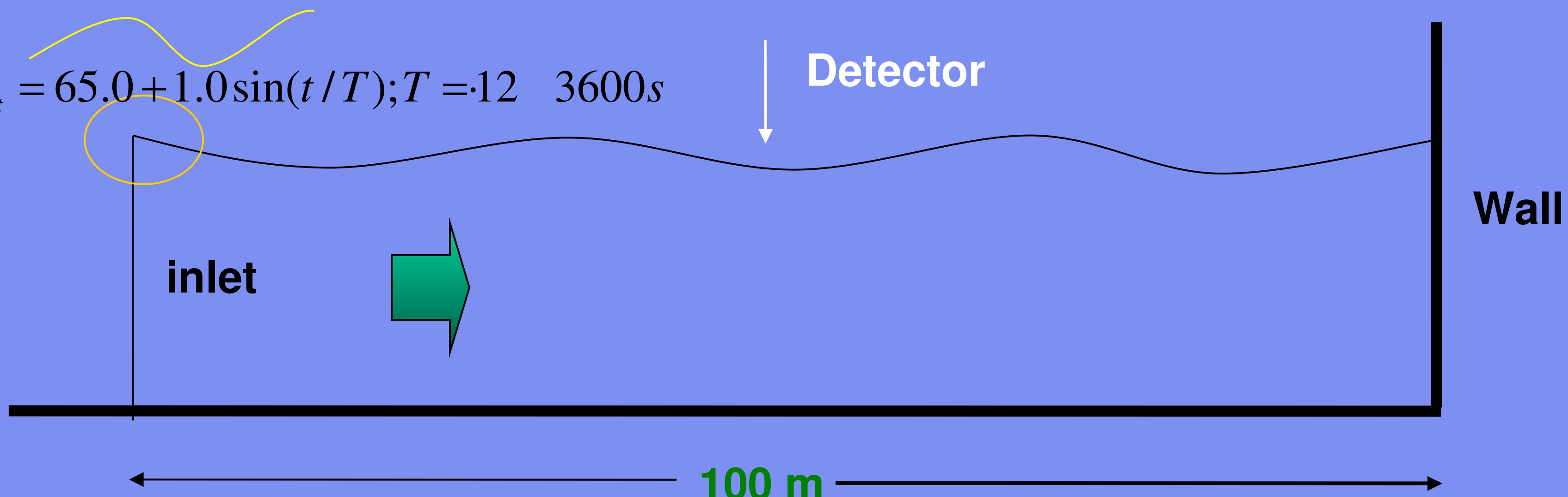
The corresponding inlet velocity: $u_b = \sqrt{g/(\eta + H_0)} \;\; \eta$

Slip boundary conditions are applied at coast and at bottom; Stress free condition on the free surface

Test case1: 2D tidal flow

To invert for free surface height at an inlet by assimilating observational data

$$\eta_{exact} = 65.0 + 1.0\sin(t/T); T = \cdot 12 \quad 3600s$$

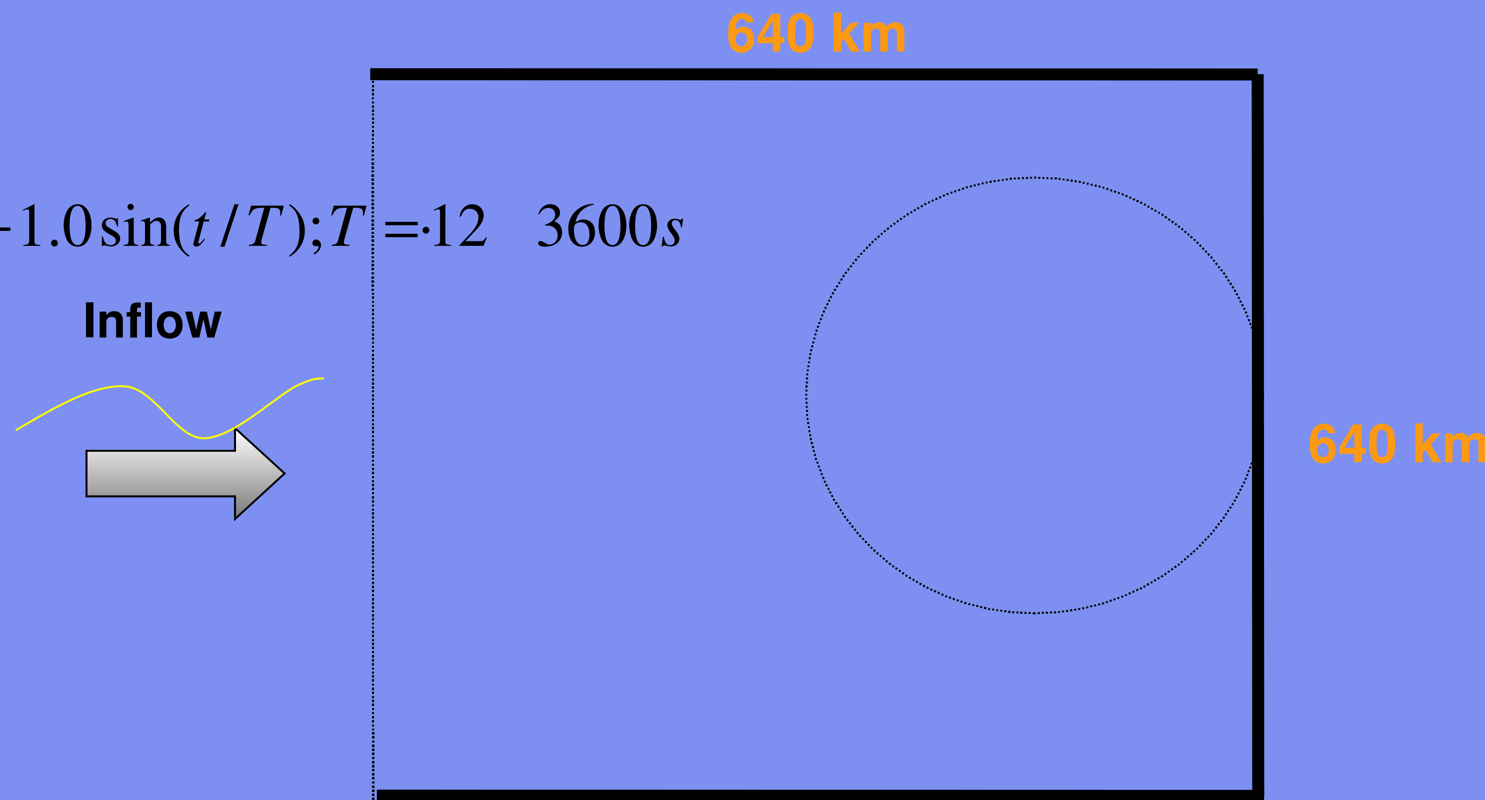


Initial guess of free surface height

$$\eta_{ini} = 65.0 + 0.5\sin(t/T); T = \cdot 12 \quad 3600s$$

Test case 2: Inversion of 3D free surface flow

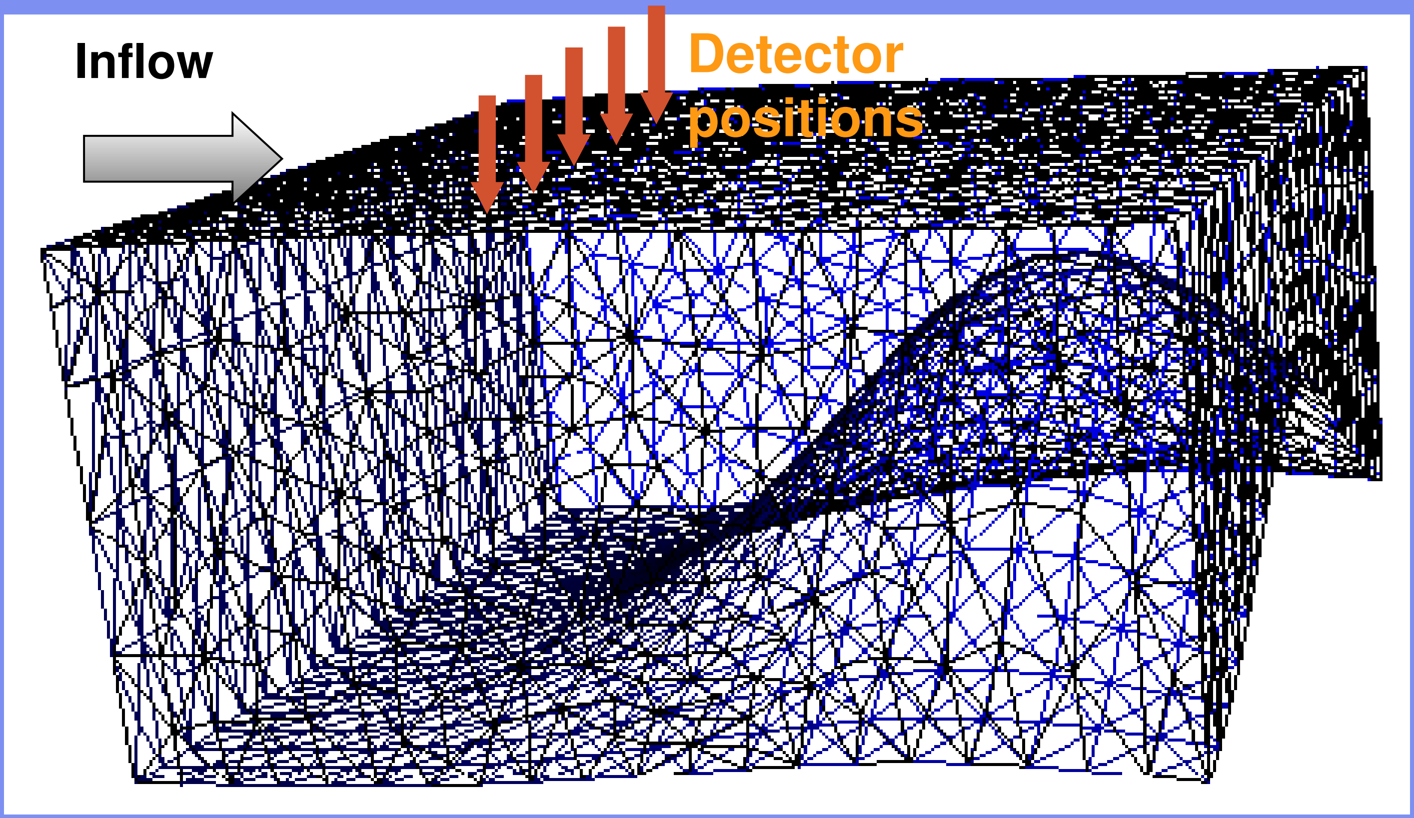
$$\eta_{exact} = 65.0 + 1.0\sin(t/T); T = \cdot 12 \quad 3600s$$



Initial guess of free surface height: $\eta_{ini} = 4(t/T - 0.5)^2 + 1; T = 12 \quad 3600s$

Seamount: Gaussian function: $h_{seamount} = 50.0e^{[(x-500000)^2 + (y-320000)^2]/2*150000^2}$

Test case 2: Inversion of 3D free surface



Rectangular gulf: 640 k*m* long and 640 k*m* wide

Seamount: Gaussian function: $$h_{seamount} = 50.0e^{[(x-500000)^2+(y-320000)^2]/2*150000^2}$$

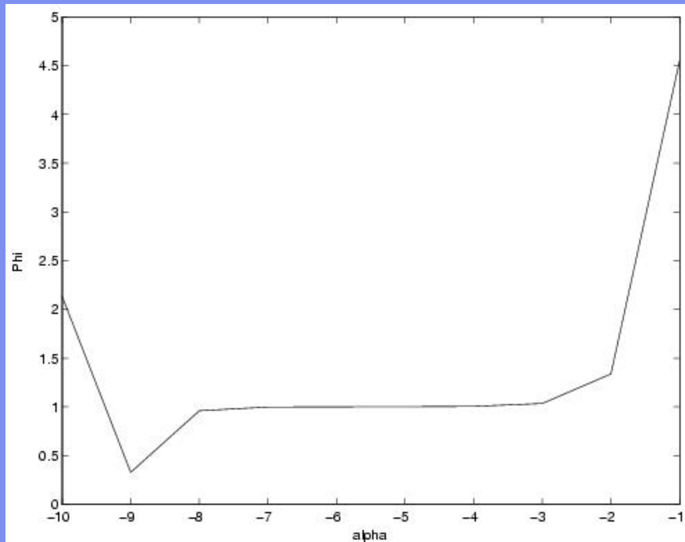
Accuracy of the adjoint model

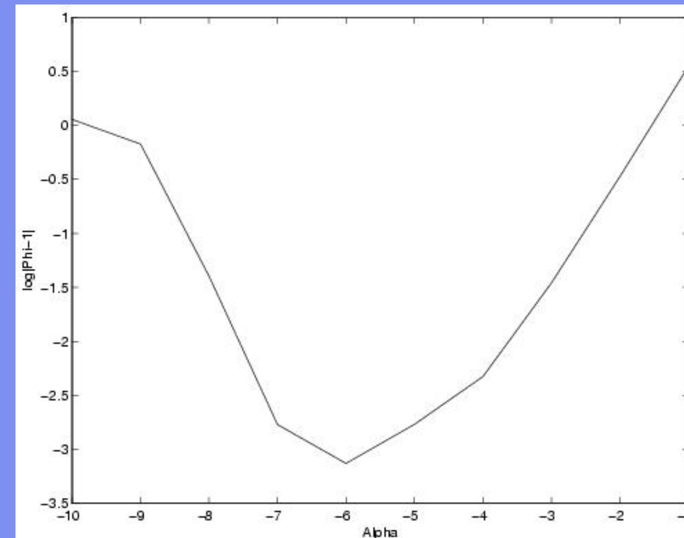
Test the consistency of the gradient (Navon, 1992)

$$\phi(\alpha) = \frac{\Im(\mathbf{m} + \alpha\mathbf{h}) - \Im(\mathbf{m})}{\alpha\mathbf{h}^T\nabla\Im(\mathbf{m})}, = 1 + O(\alpha)$$

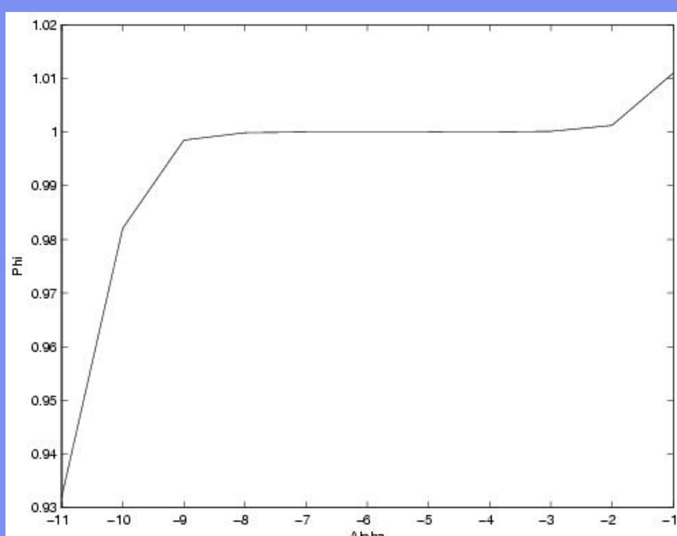
$\alpha \Rightarrow$ **0** **then** $\Phi(\alpha) \Rightarrow$ **1**

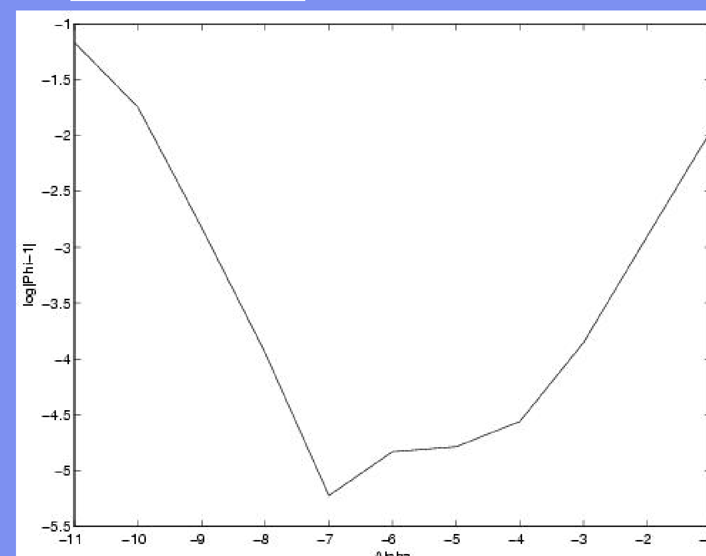
Accuracy of the adjoint model



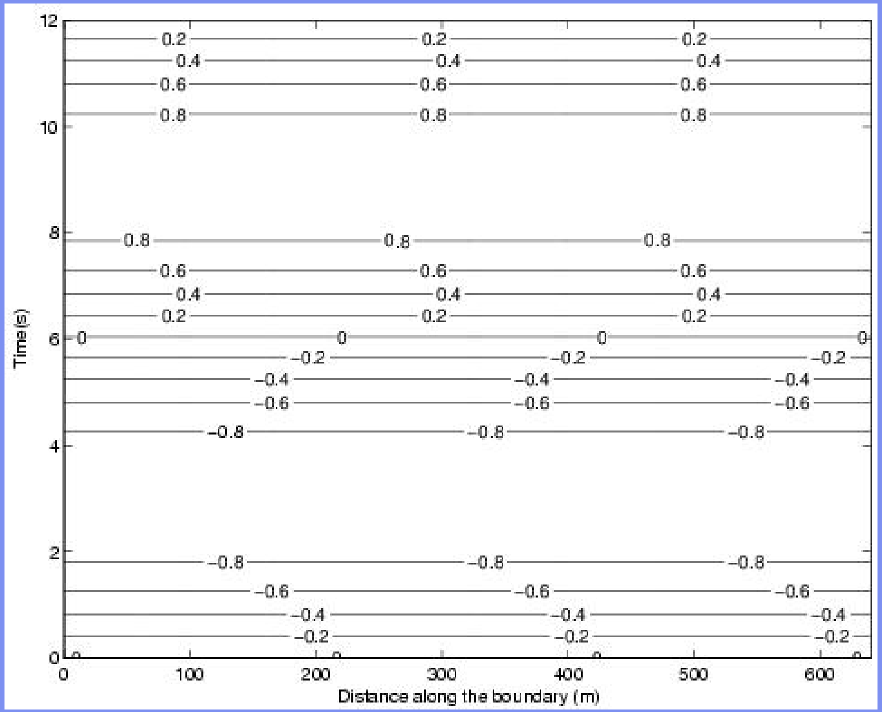


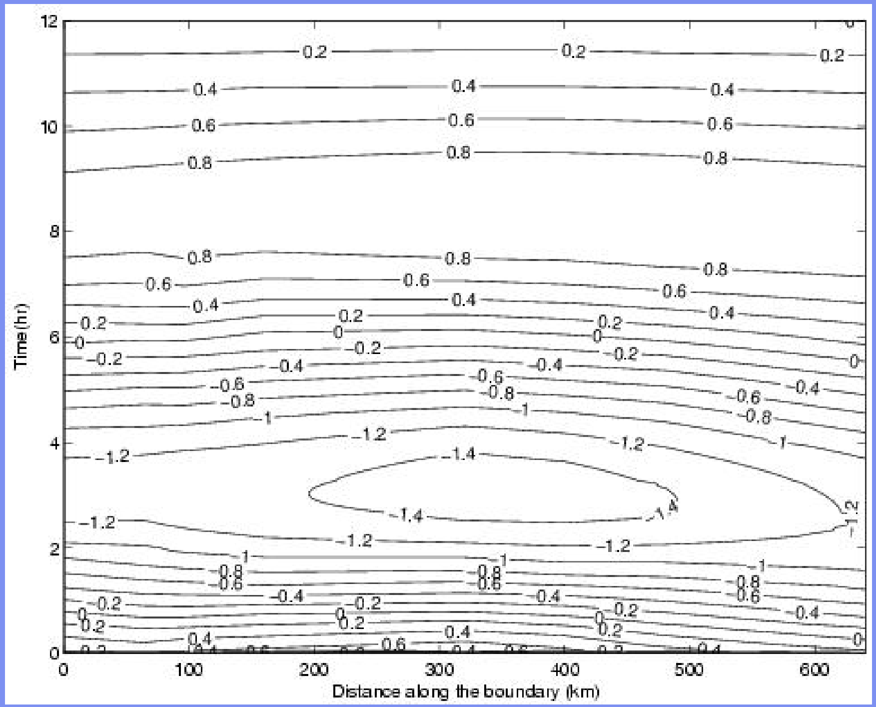
$$\log_{10}|\phi(\alpha) - 1|$$



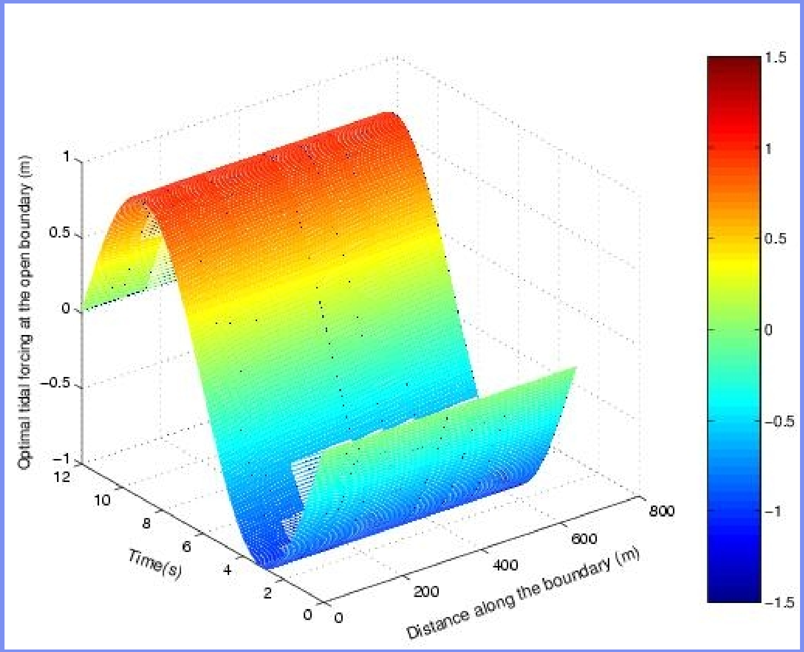


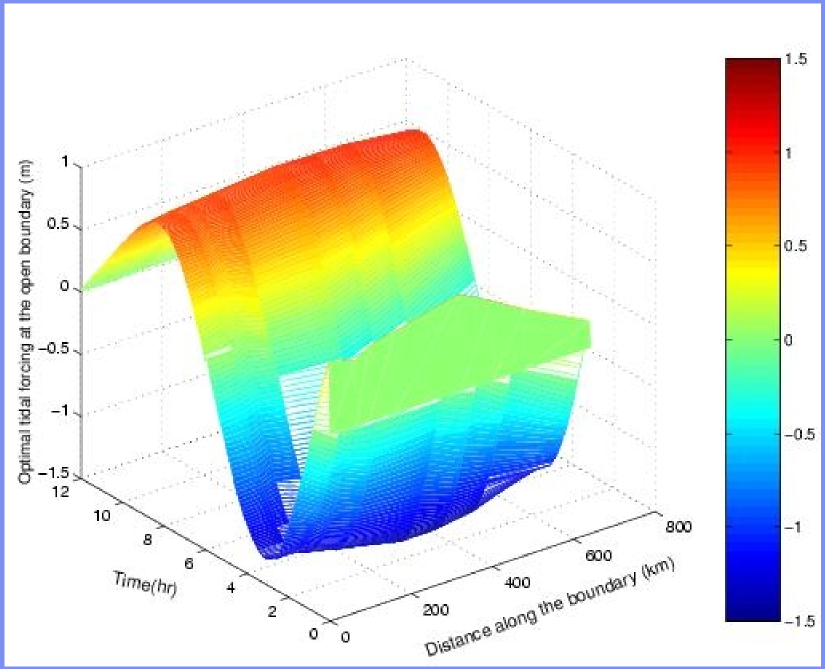
Optimised inlet tidal height.
Case 2: 3D tidal flow with a seamount



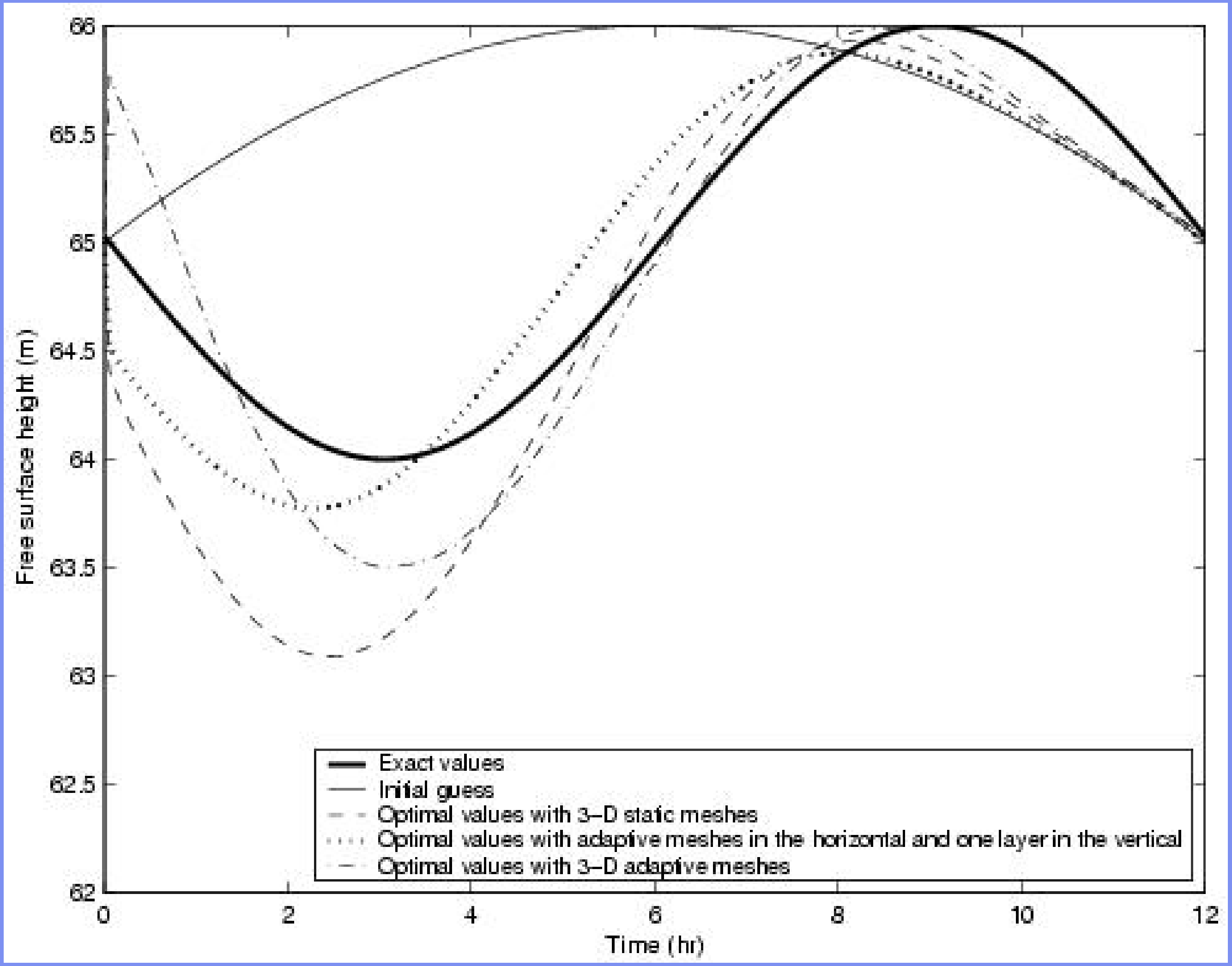


Optimised inlet tidal height.
Case 2: 3D tidal flow with a seamount





Optimised inlet tidal height at position (x= 320 km)
Case 2: 3D tidal flow with a seamount



Optimisation of uncertain parameters

Functional

$$J(\mathbf{X}, \boldsymbol{\alpha}) = \frac{1}{2}\int_{t_0}^{t_R} \langle \mathbf{W}(\mathbf{X} - \mathbf{X}^{\mathrm{obs}}), (\mathbf{X} - \mathbf{X}^{\mathrm{obs}}) \rangle \, dt + \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{g}(\boldsymbol{\alpha}),$$

Forward model

$$\frac{\partial \mathbf{X}}{\partial t} = F(\mathbf{X}, \boldsymbol{\alpha}, t).$$

Tangent Linear Model

$$\frac{\partial \delta \mathbf{X}}{\partial t} = \left(\frac{\partial F(\mathbf{X}, \boldsymbol{\alpha}, t)}{\partial \mathbf{X}}\right)\delta \mathbf{X} + \left(\frac{\partial F(\mathbf{X}, \boldsymbol{\alpha}, t)}{\partial \boldsymbol{\alpha}}\right)\delta \boldsymbol{\alpha}.$$

Adjoint model

$$-\frac{\partial \mathbf{P}}{\partial t} - \left(\frac{\partial F(\mathbf{X}, \boldsymbol{\alpha}, t)}{\partial \mathbf{X}}\right)^{\mathrm{T}} \mathbf{P} = \mathbf{W}(\mathbf{X} - \mathbf{X}^{\mathrm{obs}}),$$

Gradient

$$\nabla_{\boldsymbol{\alpha}} J = \int_{t_0}^{t_R} \left[\left(\frac{\partial F}{\partial \boldsymbol{\alpha}}\right)^{\mathrm{T}} \mathbf{P}\right] dt + \boldsymbol{\lambda}^{\mathrm{T}} \frac{\partial \mathbf{g}}{\partial \boldsymbol{\alpha}}.$$

Sensitivity analysis

Functional

$$J = \frac{1}{2}\int_t \int_\Omega U^2 d\Omega dt$$

$$\begin{aligned}
\tau_x &= WNDFAC \times C_D \rho_a (u_w - u)\sqrt{(u_w - u)^2 + (v_w - v)^2}, \\
\tau_y &= WNDFAC \times C_D \rho_a (v_w - v)\sqrt{(u_w - u)^2 + (v_w - v)^2},
\end{aligned}$$

Future Work

- Test case;
- New options;
- ??